Nonlinear large-scale data assimilation: the potential for particle filters

Peter Jan van Leeuwen

Data Assimilation Research Centre (DARC)
Department of Meteorology, University of Reading,
Reading RG6 6BB, United Kingdom
p.j.vanleeuwen@reading.ac.uk

ABSTRACT

In this chapter we will introduce the standard particle filter and show that it is degenerate, i.e. the ensemble collapses to one member for systems with dimensions larger than let's say 3. The origin of the degeneracy is found to be the large number of independent observations. We proceed by showing that instead of the original model equations a modified model equation can be used that pulls the model towards future observations. By introducing the proposal transition density we ensure that we solve the fully nonlinear data assimilation problem. Although this does greatly reduce the number of particles needed, the filter is still degenerate for systems with large dimensions. Then the so-called almost-equal-weight scheme is introduced, which is shown to solve all problems by construction. The new method is demonstrated and shown to work succesfully on a 65,000 dimensional chaotic barotropic vorticity system with observations at twice the decorrelation time scale of the model. It is concluded that particle filters can be applied to high-dimensional systems with large numbers of observations, and might be of use for numerical weather and climate prediction.

1 Introduction

In this chapter we will discuss particle filters and their use in the geosciences. A general review on the application and usefulness of particle filters in geosciences in given in Van Leeuwen (2009), see also Bocquet et al. (2010), and a general overview of particle filtering is given by the excellent book by Doucet et al, (2001). There it was shown that although interesting progress had been made until 2009, no solution for the degeneracy problem, or the curse of dimensionality had been found. In this paper we will concentrate on recent developments in using the so-called proposal transition density in solving the degeneracy problem First the basic idea behind particle filters is presented, followed by why this basic formulation can never work for large-dimensional systems. We then explore methods that do have the potential to solve the fully nonlinear data-asimilation problem, all based on methods that use the proposal density.

Finally, we will not discuss the numerous papers that explore approximations to full particle filters. A full list of this rapidly emerging field can be found in Van Leeuwen (2009), but that paper does not contain the many interesting new developments. There is room for a new review...

2 Basic Importance Sampling

The particle filters we will discuss here are based on Importance Sampling. The most straight-forward implementation is what is called Basic Importance Sampling here. (In the statistical literature one usually finds Importance Sampling described with a proposal density different from the prior model pdf.

However, for pedagogical reasons we present Importance Sampling in the following way.) Basic Importance sampling is straightforward implementation of Bayes Theorem. The idea is to represent the prior pdf by a set of particles x_i, which are delta functions centred around state vectors x_i, and from which all statistical measures of interest can be calculated, like mean, covariance etc. If one represents the prior pdf by a number of particles, or ensemble members, like in the Ensemble Kalman Filter,so

$$p(x) = \sum_{i=1}^{N} \frac{1}{N} \delta(x - x_i) \tag{1}$$

and we use this in Bayes Theorem;

$$p(x|y) = \frac{p(y|x)p(x)}{\int p(y|x)p(x)\, dx} \tag{2}$$

we find

$$p(x|y) = \sum_{i=1}^{N} w_i \delta(x - x_i) \tag{3}$$

in which the weights w_i are given by:

$$w_i = \frac{p(y|x_i)}{\sum_{j=1}^{N} p(y|x_j)} \tag{4}$$

The density $p(y|x_i)$ is the probability density of the observations given the model state x_i, which is often taken as a Gaussian:

$$p(y|x_i) = A \exp\left[-\frac{(y - H(x_i))^2}{2\sigma^2} \right] \tag{5}$$

in which $H(x_i)$ is the measurement operator, which is the model equivalent of the observation y, and σ is the standard deviation of the observation error. When more measurements are available, which might have correlated errors, the above should be the joint pdf of all these measurements.

Weighting the particles just means that their relative importance in the probability density changes. For instance, if we want to know the mean of the function $f(x)$ we now have:

$$\overline{f(x)} = \int f(x)p(x)\, dx \approx \sum_{i=1}^{N} w_i f(x_i) \tag{6}$$

Common examples for $f(x)$ are x itself, giving the mean of the pdf, and the squared deviation from the mean, giving the covariance.

Up to now, we haven't specified what x is. It can be a state vector x^n at a certain time n, or x can be a model trajectory over some time window $(0, n\Delta t)$, so $x = x^{0:n} = (x^0, x^1, ..., x^n)$ over n time steps. Here the superscript is the time index, and the subscript is the sample, or particle.

A practical way to implement the particle filter is to calculate the one time or the trajectory sequentially over time, which is where the name 'filter' comes from. The idea is to write the prior density as

$$p(x^{0:n}) = p(x^n|x^{0:n-1})p(x^{0:n-1}) \tag{7}$$

Using that the state vector evolution is Markov, i.e. to predict the future we only need the present, not the past, we can write:

$$p(x^{0:n}) = p(x^n|x^{n-1})p(x^{n-1}|p(x^{n-2})...p(x^1|x^0)p(x^0) \tag{8}$$

Before we continue it is good to realise what the so-called transition densities $p(x^n|x^{n-1})$ actually mean. Consider a model evolution equation given by:

$$x^n = f(x^{n-1}) + \beta^n \tag{9}$$

in which $\beta^n$is a random term or factor in the model equation that describes the error in the model equation. The idea is that the model is not perfect, i.e. any numerical model used in the geosciences that is used to simulate the real world has errors (and these tend to be significant!). These errors are unknown (otherwise we would include them as deterministic terms in the equations) but we assume we are able to say something about their statistics, e.g. their mean, covariance, etc. Typically one assumes the errors in the model equations are Gaussian distributed with zero mean and known covariance, but that is not always the case. To draw from such a transition density $p(x^n|x^{n-1})$ means to draw β^n from its density and evaluate the model equation given above. In fact, for normal, or Gaussian, distributed model errors β^n with covariance Q, we can write:

$$p(x^n|x^{n-1}) = N(f(x^{n-1}), Q) \tag{10}$$

Note that we assume the model errors are additive in this paper. Multiplicative model errors in which the size of the random forcing is dependent on the state x can be accounted for too, but we use additive model errors here for simplicity.

Let us now continue with Importance Sampling. If we also assume that the observations at different times are independent, which is not necessary for the formulation of the theory, but keeps the notation so much simpler, we have for the likelihood:

$$p(y^{1:n}|x^{0:n}) = p(y^n|x^n)...p(y^1|x^1) \tag{11}$$

where we used that y^j is not dependent on x^k with $j \neq k$ when x^j is known. The posterior density can now be written as:

$$\begin{aligned} p(x^{0:n}|y^{1:n}) &= \frac{p(y^{1:n}|x^{0:n})p(x^{0:n})}{p(y^{1:n})} \\ &= \frac{p(y^n|x^n)...p(y^1|x^1)p(x^n|x^{n-1})...p(x^1|x^0)p(x^0)}{p(y^n),...,p(y^1)} \\ &= \frac{p(y^n|x^n)p(x^n|x^{n-1})}{p(y^n)}...\frac{p(y^1|x^1)p(x^1|x^0)p(x^0)}{p(y^1)} \end{aligned} \tag{12}$$

Realising that the last ratio in this equation is actually equal to $p(x^{0:1}|y^1)$ we find the following sequential relation:

$$p(x^{0:n}|y^{0:n}) = \frac{p(y^n|x^n)p(x^n|x^{n-1})}{p(y^n)} p(x^{0:n-1}|y^{1:n-1}) \tag{13}$$

This expression allows us to find the full posterior with the following sequential scheme(see figure 1):

1 Sample N particles x_i from the initial model probability density $p(x^0)$, in which the superscript 0 denotes the time index.

2 Integrate all particles forward in time up to the measurement time. In probabilistic language we denote this as: sample from $p(x^n|x_i^{n-1})$ for each i,
i.e. for each particle x_i run the model forward from time $n-1$ to time n using the nonlinear model equations. The stochastic nature of the forward evolution is implemented by sampling from the density that describes the random forcing of the model.

3 Calculate the weights according to (4), normalise them so their sum is equal to 1, and attach these weights to each corresponding particle. Note that the particles are not modified, only their relative weight is changed!

4 Increase n by one and repeat 2 and 3 until all observations have been processed.

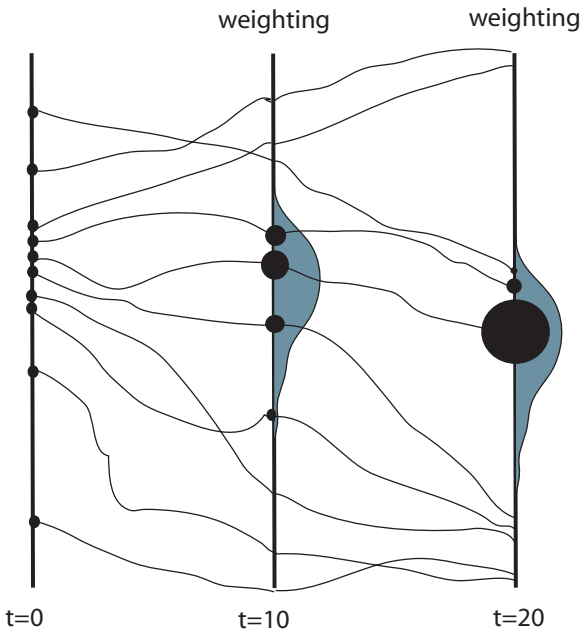

Figure 1: The standard particle filter with Importance Sampling. The model variable runs along the vertical axis, the weight of each particle corresponds to the size of the bullets on this axis. The horizontal axis denotes time, with observations at a time interval of 10 time units. All particles have equal weight at time 0. At time 10 the likelihood is displayed together with the new weights of each particle. At time 20 only 2 members have weights different from zero: the filter has become degenerate.

The good thing about importance sampling is that the particles are not modified, so that dynamical balances are not destroyed by the analysis. The bad thing about importance sampling is that the particles are not modified, so that when all particles move away from the observations they are not pulled back to the observations. Only their relative weights are changed.

Before we continue, it is stressed how simple this scheme is compared to traditional methods like 3- or 4DVar and (Ensemble) Kalman filters. The success of these scheme depends heavily on the accuracy and error covariances of the model state vector. In 3- and 4DVar this leads to complicated covariance structures to ensure balances etc. In Ensemble Kalman filters artificial tricks like covariance inflation and localisation are needed to get good results in high dimensional systems. Particle filters do not have these difficulties.

However, there is (of course) a drawback. Even if the particles manage to follow the observations in time, the weights will differ more and more. Application to even very low-dimensional systems shows that after a few analysis steps one particle gets all the weight, while all other particles have very low weights (see figure 1, at $t = 20$). That means that the statistical information in the ensemble becomes too low to be meaningful. This is called *filter degeneracy*. It has given importance sampling a low profile until resampling was invented, see the next section.

3 Reducing the variance in the weights

Several methods exists to reduce the variance in the weights, and we discuss Sequential Importance Resampling here. See Van Leeuwen(2009) for other methods. In resampling methods the posterior ensemble is resampled so that the weights become more equal (Gordon et al., 1993). In the next section

after this one methods are discussed that do change the positions of the prior particles in state space to improve the likelihood of the particles, i.e. to bring them closer to the observations before the weighting with the likelihood is applied.

3.1 Resampling

The idea of resampling is simply that particles with very low weights are abandoned, while multiple copies of particles with high weight are kept for the posterior pdf in the sequential implementation. In order to restore the total number of particles N, identical copies of high-weight particles are formed. The higher the weight of a particle the more copies are generated, such that the total number of particles becomes N again. Sequential Importance Re-sampling (SIR) does the above, and makes sure that the weights of all posterior particles are equal again, to $1/N$.

Sequential Importance Re-sampling is identical to Basic Importance Sampling but for a resampling step after the calculation of the weights. The 'flow chart' reads (see figure 2):

1 Sample N particles x_i from the initial model probability density $p(x^0)$.

2 Integrate all particles forward in time up to the measurement time (so, sample from $p(x^n|x_i^{n-1})$ for each i)

3 Calculate the weights according to (4) and attach these weights to each corresponding particle. Note that the particles are not modified, only their relative weight is changed!

4 Re-sample the particles such that the weights are equal to $1/N$.

5 Repeat 2, 3 and 4 sequentially until all observations have been processed.

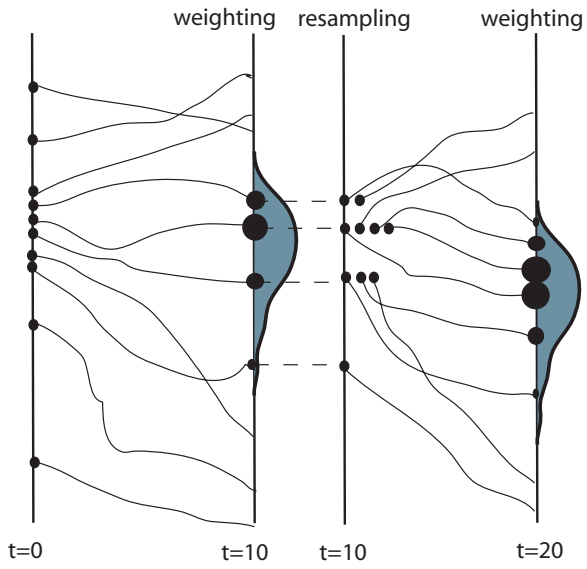

Figure 2: The Particle Filter with Resampling, also called Sequential Importance Resampling. The model variable runs along the vertical axis, the weight of each particle corresponds to the size of the bullets on this axis. The horizontal axis denotes time, with observations at a time interval of 10 time units. All particles have equal weight at time zero. At time 10 the particles are weighted according to the likelihood, and resampled to obtain an equal-weight ensemble.

It is good to realise that the resampling step destroys the smoother character of the method. All particles that are not chosen in the resampling scheme are lost, and their evolution is broken. So the smoother

estimate is build of of lesser and lesser particles over time, until it consists of only one particle, loosing again all statistical meaning.

The resampling can be performed in many ways, and we discuss the most used.

1) *Probabilistic resampling*

   Most straightforward is to directly sample randomly from the density given by the weights. Since this density is discrete and one-dimensional this is an easy task. However, due to the random character of the sampling, so-called sampling noise is introduced. Note that this method is actually generalised Bernoulli for those versed in sampling techniques..

2) *Residual Sampling*

   To reduce the sampling noise Residual Sampling can be applied. In this re-sampling method all weights are multiplied with the ensemble size N. Then n copies are taken of each particle i in which n is the integer part of Nw_i. After obtaining these copies of all members with $Nw_i \geq 1$, the integer parts of Nw_i are subtracted from Nw_i. The rest of the particles needed to obtain ensemble size N are than drawn randomly from this resulting distribution.

3) *Stochastic Universal Sampling*

   While Residual Sampling reduces the sampling noise, it can bee shown that Stochastic Universal Sampling has lowest sampling noise. In this method all weights are put after each other on the unit interval $[0,1]$. Then a random number is drawn from a uniform density on $[0,1/N]$, and N line pieces starting from the random number, and with interval length $1/N$ are laid on the line $[0,1]$. A particle is chosen when one of the end points of these line pieces falls in the weight bin of that particle. Clearly, particles with high weights span an interval larger than $1/N$ and will be chosen a number of times, while small weight particles have a negligible change of being chosen.

3.2 Is resampling enough?

Snyder et al. (2008) prove that resampling will not be enough to avoid filter collapse, i.e. one particle gets all the weight, and the others get weight zero. See also the contribution by Chris Snyder to this symposium. Their argument is quite involved and makes a few assumptions that cannot be fully justified, although the numerical experiments performed fit the theory perfectly. These authors argue that the number of particles should grow exponentially with the dimension of the system. Below we show that it is the number of independent observations that brings about the ensemble collapse, and not so much the dimension of the system. To do that we first discuss the idea of the proposal density in particle filtering in the next session.

4 Improvement of the likelihood: exploring the proposal density

Related to decreasing the variance of the weights is to make sure that all model integrations end up close to the new observations. First we discuss the application of a proposal density that allows one to sample from a density that is conditioned on these observations, so it is much closer to the observations than the prior density. As an example the EnKF is chosen as the proposal density. Then we discuss methods that modify the model equations. The so-called 'optimal proposal density' is discussed, and it is shown it is not optimal, followed by a scheme that does solve the degeneracy problem by construction.

4.1 The proposal density: the basic idea

We are now to discuss a very interesting property of particle filters that has received little attention in the geophysical community. We start from Bayes, i.e. eq (13):

$$p(x^{0:n}|y^{0:n}) = \frac{p(y^n|x^n)p(x^n|x^{n-1})}{p(y^n)} p(x^{0:n-1}|y^{1:n-1}) \tag{14}$$

To simplify the analysis, and since we concentrate on a filter here, let us first integrate out the past, to get:

$$p(x^n|y^{0:n}) = \frac{p(y^n|x^n)}{p(y^n)} \int p(x^n|x^{n-1})p(x^{n-1}|y^{1:n-1})\, dx^{n-1} \tag{15}$$

This expression does not change when we multiply and divide by a so-called proposal transition density $q(x^n|x^{n-1},y^n)$, so:

$$p(x^n|y^{0:n}) = \frac{p(y^n|x^n)}{p(y^n)} \int \frac{p(x^n|x^{n-1})}{q(x^n|x^{n-1},y^n)} q(x^n|x^{n-1},y^n)p(x^{n-1}|y^{1:n-1})\, dx^{n-1} \tag{16}$$

As long as the support of $q(x^n|x^{n-1},y^n)$ is equal to or larger than that of $p(x^n|x^{n-1})$ we can always do this. This last condition makes sure we don't divide by zero. Let us now assume that we have an equal-weight ensemble of particles from the previous analysis at time $n-1$, so

$$p(x^{n-1}|y^{1:n-1}) = \sum_{i=1}^{N} \frac{1}{N} \delta(x^{n-1} - x_i^{n-1}) \tag{17}$$

Using this in the equation above gives:

$$p(x^n|y^{0:n}) = \sum_{i=1}^{N} \frac{1}{N} \frac{p(y^n|x^n)}{p(y^n)} \frac{p(x^n|x_i^{n-1})}{q(x^n|x_i^{n-1},y^n)} q(x^n|x_i^{n-1},y^n) \tag{18}$$

As a last step, we run the particles from time $n-1$ to n, i.e. we sample from the transition density. However, instead of drawing from $p(x^n|x_i^{n-1})$, so running the original model, we sample from $q(x^n|x_i^{n-1},y^n)$, so from a modified model. Let us write this modified model as

$$x^n = g(x^{n-1},y^n) + \hat{\beta}^n \tag{19}$$

so that we can write for the transition density, assuming $\hat{\beta}^n$ is Gaussian distributed with covariance $\hat{Q}$:

$$q(x^n|x^{n-1},y^n) = N(g(x^{n-1},y^n),\hat{Q}) \tag{20}$$

Drawing from this density leads to:

$$p(x^n|y^{0:n}) = \sum_{i=1}^{N} \frac{1}{N} \frac{p(y^n|x_i^n)}{p(y^n)} \frac{p(x_i^n|x_i^{n-1})}{q(x_i^n|x_i^{n-1},y^n)} \delta(x^n - x_i^n) \tag{21}$$

so the posterior pdf at time n can be written as:

$$p(x^n|y^{1:n}) = \sum_{i=1}^{N} w_i \delta(x^n - x_i^n) \tag{22}$$

with weights w_i given by:

$$w_i = \frac{1}{N} \frac{p(y^n|x_i^n)}{p(y^n)} \frac{p(x_i^n|x_i^{n-1})}{q(x_i^n|x_i^{n-1},y^n)} \tag{23}$$

We recognise the first factor in this expression as the likelihood, and the second as as a factor related to using the proposal transition density instead of the original transition density to propagate from time $n-1$ to n, so it is related to the use of the proposed model instead of the original model. Note that because the factor $1/N$ and $p(y^n)$ are the same for each particle and we are only interested in relative weights, we will drop them from now on, so

$$w_i = p(y^n|x_i^n)\frac{p(x_i^n|x_i^{n-1})}{q(x_i^n|x_i^{n-1},y^n)} \tag{24}$$

4.2 Example: the EnKF as proposal

As an example we will explore this technique with the Gaussian of the EnKF as the proposal density. First we have to evaluate the prior transition density. Since we know the starting point of the simulation, x_i^{n-1}, and its end point, the posterior EnKF sample x_i^n, and we know the model equation, written formally as:

$$x_i^n = f(x_i^{n-1}) + \beta_i^n \tag{25}$$

we can determine β_i^n from this equation directly. We also know the distribution from which this β_i^n is supposed to be drawn, let us say a Gaussian with zero mean and covariance Q. We then find for the transition density:

$$p(x_i^n|x_i^{n-1}) \propto \exp\left[-1/2\left(x_i^n - f(x_i^{n-1})\right)Q^{-1}\left(x_i^n - f(x_i^{n-1})\right)\right] \tag{26}$$

This will give us a number for each $[x_i^{n-1},\, x_i^n]$ combination.

Let us now calculate the proposal density $q(x_i^n|x_i^{n-1},y^n)$. This depends on the ensemble Kalman filter used. For the Ensemble Kalman filter with perturbed observations the situation is as follows. Each particle in the updated ensemble is connected to those before analysis as:

$$x_i^n = x_i^{n,old} + K^e\left(y + \varepsilon_i - H(x_i^{n,old})\right) \tag{27}$$

in which ε_i is the random error drawn from $N(0,R)$ that has to be added to the observations in this variant of the ensemble Kalman filter. K^e is the ensemble Klman gain, i.e. the Kalman gain using the prior error covariance calculated from the prior ensemble. The particle prior to the analysis comes from that of the previous time step through the stochastic model:

$$x_i^{n,old} = f(x^{n-1}) + \beta_i^n \tag{28}$$

Combining these two gives:

$$x_i^n = f(x_i^{n-1}) + \beta_i^n + K^e\left(y + \varepsilon_i - H(x_i^{n-1}) - H(\beta_i^n))\right) \tag{29}$$

or

$$x_i^n = f(x_i^{n-1}) + K^e\left(y - H(f(x_i^{n-1}))\right) + (1 - K^eH)\beta_i^n + K^e\varepsilon_i \tag{30}$$

assuming that H is a linear operator. The right-hand side of this equation has a deterministic and a stochastic part. The stochastic part provides the transition density going from x_i^{n-1} to x_i^n. Assuming both model and observation errors to be Gaussian distributed and independent we find for this transition density:

$$q(x_i^n|x_i^{n-1}y^n) \propto \exp\left[-1/2\,(x_i^n - \mu_i^n)^T\,\Sigma_i^{-1}\,(x_i^n - \mu_i^n)\right] \tag{31}$$

in which μ_i^n is the deterministic 'evolution' of x, given by:

$$\mu_i^n = f(x_i^{n-1}) + K^e\left(y - H(x_i^{n-1})\right) \tag{32}$$

and the covariance Σ_i is given by:

$$\Sigma_i = (1 - K^e H) Q (1 - K^e H)^T + K^e R K^{eT} \tag{33}$$

where we assumed that the model and observation errors are uncorrelated. It should be realized that x_i^n does depend on all $x_j^{n,old}$ via the Kalman gain, that involves the error covariance P^e. Hence we have calculated $q(x_i^n|P^e, x_i^{n-1}, y^n)$ instead of $q(x_i^n|x_i^{n-1}, y^n)$, in which P^e depends on all other particles. The reason why we ignore the dependence on P^e is that in case of an infinitely large ensemble P^e would be a variable that depends only on the system, not on specific realizations of that system. This is different from the terms related to x_i^n, that will depend on the specific realization for β_i^n even when the ensemble size is 'infinite'. (Hence another approximation related to the finite size of the ensemble comes into play here and at this moment it is unclear how large this approximation error is.)

The calculation of $p(x^n|x^{n-1})$ and $q(x_i^n|x_i^{n-1}y^n)$ look like very expensive operations. By realizing that Q and R can be obtained from the ensemble of particles, computationally efficient schemes can easily be derived.

We can now determine the full new weights. Since the normalization factors for the transition and the posterior densities are the same for all particles the weights are easily calculated. The procedure now is as follows (see figure 3):

1 Run the ensemble up to the observation time

2 Perform a (local) EnKF analysis of the particles

3 Calculate the proposal weights $w_i^* = p(x_i^n|x_i^{n-1})/q(x_i^n|x_i^{n-1}y^n)$

4 Calculate the likelihood weights $w_i = p(y^n|x_i^n)$

5 Calculate the full relative weights as $w_i = w_i * w_i^*$ and normalize them.

6 Resample

It is good to realize that the EnKF step is only used to draw the particles close to the observations. This means that when the weights are still varying too much, one can do the EnKF step with much lower observational errors. This might look like overfitting but it is not since the only thing we do in probabilistic sense is to generate particles to those positions in state space where the likelihood is large.

Finally, other variants of the EnKF, like the adjusted and the transform variants can be used too, as detailed in Van Leeuwen (2009). The efficiency of using the EnKF as proposal is under debate at the moment. The conclusions so far seem to be that using the EnKF as proposal in high-dimensional systems does not work. What has not been tested, however, is to use EnKF proposals with smaller observation matrix R, and more possibilities are still open.

4.3 Modifying the model equations

So far we have discussed proposal density applications in which the model equations were not changed directly. However, much more efficient scheme's can be derived that change the model equations such that each particle is pulled towards the future observations at each time step. By keeping track of the weights associated with this it can be assured that the correct problem is solved, and the particles are random samples from the posterior pdf.

As mentioned before, the idea of the proposal transition density is that we draw samples from that density instead of from the original model. Furthermore, these samples can be dependent on the future

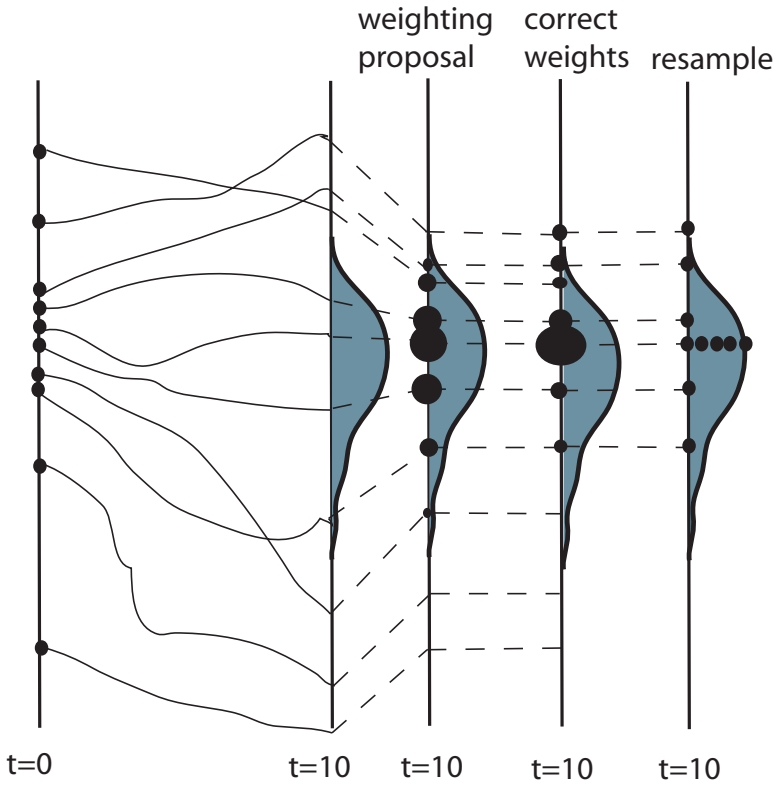

Figure 3: The particle filter with proposal density. The model variable runs along the vertical axis, the weight of each particle corresponds to the size of the bullets on this axis. The horizontal axis denotes time, with observations at a time interval of 10 time units. All particles have equal weight at time zero. At time 10 the particles are brought closer to the observations by using e.g. the EnKF. Then they are weighted with the likelihood and these weights are corrected for the artificial EnKF step.

observations. To see how this works, let us write the stochastic model equation as:

$$x_i^n = f(x_i^{n-1}) + \beta_i^n \tag{34}$$

First we have to understand how this equation is related to the transition density $p(x_i^n|x_i^{n-1})$. The probability to end up in x_i^n starting from x_i^{n-1} is related to β_i^n. For instance, if $\beta_i^n = 0$, so no model error, a perfect model, this probability is 1 if the x_i^n, x_i^{n-1} pair fulfils the perfect model equations, and zero otherwise. So, in this case $p(x_i^n|x_i^{n-1})$. would be a delta function centred on $f(x_i^{n-1})$. However, the more realistic case is that the model error is nonzero. The transition density will now depend on the distribution of the stochastic random forcing. Assuming Gaussian random forcing with mean zero and covariance Q, so $\beta_i^n \sim N(0,Q)$, we find

$$p(x_i^n|x_i^{n-1}) \propto N(f(x_i^{n-1}),Q) \tag{35}$$

As mentioned above, we will not use the normal model equation for each particle, but a modified model equation, one that 'knows' about future observations, and actually draws the model to those observations. Perhaps the simplest example is to add a term that relaxes the model to the future observation, like

$$x_i^n = f(x_i^{n-1}) + \beta_i^n + K^n(y^{n+m} - H(x_i^{n-1}) \tag{36}$$

in which $n+m$ is the next observation time. Note that the observation operator H does not contain any model integrations, it is just the evaluation of x_i^{n-1} in observation space. The reason is simple, we

don't have x_i^{n+m} yet. Clearly, each particle i will now be pulled towards the future observations, with relaxation strength related to matrix K^n. In principle, we are free to choose K^n, but it is reasonable to assume that it is related to the error covariance of the future observation R, and that of the model equations Q. We will show possible forms in the examples discussed later.

With the simple relaxation, or other techniques, we have ensured that all particles end up closer to the observations. But we can't just alter the model equations, we have to compensate for this trick. This is why the proposal density turns up in the weights. Each time step the weight of each particle changes with

$$w_i^n = \frac{p(x_i^n|x_i^{n-1})}{q(x_i^n|x_i^{n-1},y^n)} \tag{37}$$

between observation times. This can be calculated in the following way. Using the modified model equations, we know x_i^{n-1} for each particle, that was our starting point, and also x_i^n. So, assuming the model errors are Gaussian distributed, this would become

$$p(x_i^n|x_i^{n-1}) \propto \exp\left[-\frac{1}{2}\left(x_i^n - f(x_i^{n-1})\right)^T Q^{-1}\left(x_i^n - f(x_i^{n-1})\right)\right] \tag{38}$$

The proportionality constant is not of interest since it is the same for each particle, and drops out when the relative weights of the particles are calculated. Note that we have all ingredients to calculate this, and $p(x_i^n|x_i^{n-1})$ is just a number.

For the proposal transition density we use the same argument, to find:

$$\begin{aligned} q(x_i^n|x_i^{n-1}y^n) &\propto \exp\left[-\frac{1}{2}\left(x_i^n - f(x_i^{n-1}) - K^n(y^n - H(x^{n-1}))\right)^T Q^{-1}\left(x_i^n - f(x_i^{n-1}) - K^n(y^n - H(x^{n-1}))\right)\right] \\ &= \exp\left[-\frac{1}{2}\beta_i^{nT} Q^{-1}\beta_i^n\right] \end{aligned} \tag{39}$$

Again, since the did choose β to propagate the model state forward in time, we can calculate this and it is just a number. In this way, any modified equation can be used, and we know, at least in principle, how to calculate the appropriate weights.

4.4 The 'Optimal proposal density'

In the literature the so-called 'optimal proposal density' is described (e.g. Doucet et al, 2000). It is argued that taking $q(x^n|x^{n-1},y^n) = p(x^n|x^{n-1},y^n)$ results in optimal weights. However, it is easy to show that this is not the case. Assume observations every time step, and a resampling scheme at every time step, so that a equal-weighted ensemble of particles is present at time $n-1$. Furthermore, assume that model errors are Gaussian distributed $N(0,Q)$ and observation errors are Gaussian distributed according to $N(0,R)$. First, using the definition of conditional densities we can write:

$$p(x^n|x^{n-1},y^n) = \frac{p(y^n|x^n)p(x^n|x^{n-1})}{p(y^n|x^{n-1})} \tag{40}$$

where we used $p(y^n|x^n,x^{n-1}) = p(y^n|x^n)$. Using this proposal density gives posterior weights:

$$\begin{aligned} w_i &= p(y^n|x_i^n)\frac{p(x_i^n|x_i^{n-1})}{q(x_i^n|x_i^{n-1},y^n)} \\ &= p(y^n|x_i^n)\frac{p(x_i^n|x_i^{n-1})}{p(x_i^n|x_i^{n-1},y^n)} \\ &= p(y^n|x_i^{n-1}) \end{aligned} \tag{41}$$

The latter can be expanded as:

$$w_i = \int p(y^n, x^n | x^{n-1})\, dx^n = \int p(y^n|x^n) p(x^n|x^{n-1})\, dx^n \tag{42}$$

in which we again used $p(y^n|x^n, x^{n-1}) = p(y^n|x^n)$. Using the Gaussian assumptions mentioned above (note, the state is never assumed to be Gaussian), we can perform the integration to obtain:

$$w_i \propto \exp\left[-\frac{1}{2}\left(y^n - Hf(x_i^{n-1})\right)^T (HQH^T + R)^{-1}\left(y^n - Hf(x_i^{n-1})\right)\right] \tag{43}$$

Note that we have just calculated the probability density of $p(y^n|x_i^{n-1})$.

To estimate the order of magnitude of the first two moments of the distribution of $y - Hf(x_i^{n-1})$ it is expanded to $y - Hx_t^n + H\left(x_t^n - f(x_i^{n-1})\right)$ in which x_t^n the true state at time n. If we now use $x_t^n = f(x_t^{n-1}) + \beta_t^n$ this can be expanded further as $y - Hx_t^n + H\left(f(x_t^{n-1}) - f(x_i^{n-1})\right) + H\beta_t^n$. To proceed we make the following restrictive assumptions that will nevertheless allow us to obtain useful order-of-magnitude estimates. Let us assume that both the observation errors R and the observed model errors HQH^T are uncorrelated, with variances V_y and V_β, respectively, to find:

$$-\log(w_i) = \frac{1}{2(V_\beta + V_y)} \sum_{j=1}^{M} \left[y_j - H_j x_t^n + H_j \beta_t^n + H_j\left(f(x_t^{n-1}) - f(x_i^{n-1})\right)\right]^2 \tag{44}$$

The variance of w_i arises from varying ensemble index i. Clearly the first three terms are given, and we introduce the constant $\gamma_j = y_j^n - H_j x_t^n + H_j \beta_t^n$. To proceed with our order of magnitude estimate we assume that the model can be linearised as $F(x_i^{n-1}) \approx A x_i^{n-1}$., leading to:

$$-\log(w_i) = \frac{1}{2(V_\beta + V_y)} \sum_{j=1}^{M} \left[\gamma_j + H_j A(x_t^{n-1} - x_i^{n-1})\right]^2 \tag{45}$$

A following step in our order of magnitude estimate is to assume $x_t^{n-1} - x_i^{n-1}$ to be Gaussian distributed. In that case the expression above is non-central χ^2_M distributed apart from a constant. This constant comes from the variance of $\gamma_j + H_j A(x_t^{n-1} - x_i^{n-1})$, which is equal to $H_j A P^{n-1} A^T H_j^T = V_x$, in which P^{n-1} is the covariance of the model state at time $n-1$. Hence we find:

$$-\log(w_i) = \frac{V_x}{2(V_\beta + V_y)} \sum_{j=1}^{M} \frac{\left[\gamma_j + H_j A(x_t^{n-1} - x_i^{n-1})\right]^2}{V_x} \tag{46}$$

Apart from the constant in front the expression above is non-central χ^2_M distributed with variance $a^2 2(M + 2\lambda)$ where $a = V_x/(2(V_\beta + V_y)$ and $\lambda = (\sum_j \gamma_j^2)/V_x$.

We can estimate λ by realising that for a large enough number of observations we expect $\sum_j (y_j^n - H_j x_t^n)^2 \approx M V_y$, and $\sum_j (y_j^n - H_j x_t^n) \approx 0$. Furthermore, when the dimension of the system under study is large we expect $\sum_j (H_j \beta_t^n)^2 \approx M V_\beta$. Combining all these estimates we find that the variance of $-\log(w_i)$ can be estimated as

$$\frac{M}{2}\left(\frac{V_x}{V_\beta + V_y}\right)^2 \left(1 + 2\left(\frac{V_\beta + V_y}{V_x}\right)\right) \tag{47}$$

This expression shows that the only way to keep the variance of $-\log(w_i)$ low when the number of independent observations M is large is to have a very small variance in the ensemble: $V_x \approx (V_\beta + V_y)/M$. Clearly, in when the number of observations is large (10 million in typical meteorological applications), this is not very realistic.

It should be mentioned that a large variance of $-\log(w_i)$ does not necessarily mean that the weights will be degenerate because the large variance could be due to a few outliers. However, we have shown

that $-\log(w_i)$ is approximately non-central χ^2_M distributed for a linear model, so the large variance is not due to outliers but intrinsic in the sampling from such a distribution. Furthermore, there is no reason to assume that this variance will behave better for nonlinear models, especially because we didn't make any assumptions on the divergent or contracting characteristics of the linear model.

From this analysis we learn two things: it is the number of independent observations that determines the degeneracy of the filter, and the optimal proposal density cannot be used in systems with a very large number of independent accurate observations.

4.5 The almost equal weights scheme

Unfortunately, exploring the proposal density by simply nudging will not avoid degeneracy in high-dimensional systems with a large number of observations. Also more complicated scheme's, such as running a 4DVar on each particle, which is essentially what Chorin et al. (2009) propose will lead to strongly varying weights for the particles. The reason for the latter is related to the 'optimal' proposal-density scheme presented in the previous section. A 4DVar solves for the maximum of $p(x_i^n|x_i^{n-1},y^m)$, in which $m > n$. This is a deterministic move, and adding a random number from the the posterior will give a consistent particle. However, the weights of the particles will be similar to the expression given for the 'optimal' proposal density scheme, so degenerate. We have to do something more optimal.

Making sure that all particles end up relatively close to the observations still does not mean that the weights will not vary wildly in large-dimensional systems. A new ingredient is that we ensure that all posterior weights are almost equal. This consists of two stages: first perform a deterministic time step with each particle that ensures that most of the particles have equal weight, and then add a very small random step to ensure that Bayes theorem is satisfied, see Van Leeuwen (2010, 2011) for details. There are again infinitely many ways to do this.

For the first stage we write down the weight for each particle using only a deterministic move, so ignoring the proposal density q for the moment:

$$-\log w_i = -\log w_i^{rest} + \frac{1}{2}(y^n - Hx_i^n)^T R^{-1}(y^n - Hx_i^n) + \frac{1}{2}(x_i^n - f(x_i^{n-1}))^T Q^{-1}(x_i^n - f(x_i^{n-1})) \quad (48)$$

in which w_i^{rest} is the weight accumulated over the previous time steps between observations, so the p/q factors from each time step. If H is linear, which is not essential but as we will assume for simplicity here, this is a quadratic equation in the unknown x_i^n. All other quantities are given. We calculate the minimum of this function for each particle i, which is simply given by

$$-\log w_i = -\log w_i^{rest} + \frac{1}{2}\left(y^n - Hf(x_i^{n-1})\right)^T (HQH^T + R)^{-1}\left(y^n - Hf(x_i^{n-1})\right) \quad (49)$$

For N particles this given rise to N minima. Next, we determine a target weight as the weight that 80% of the particles can reach, i.e. 80% of the minimum $-\log w_i$ is smaller than the target value. Define a quantity $C = -\log w_{target}$, and solve for each particle with a minimum weight larger than the target weight

$$C = -\log w_i^{rest} + \frac{1}{2}\left(y^n - Hf(x_i^{n-1})\right)^T (HQH^T + R)^{-1}\left(y^n - Hf(x_i^{n-1})\right) \quad (50)$$

So now we have found the positions of the new particles x_i^n such that all have equal weight. The particles that have a larger minimum than C will come back into the ensemble via a resampling step, to be discussed later.

The equation above has an infinite number of solutions for dimensions larger than 1. To make a choice we assume

$$x_i^n = f(x_i^{n-1}) + \alpha_i K[y^n - Hf(x_i^{n-1})] \quad (51)$$

in which $K = QH^T(HQH^T + R)^{-1}$, Q is the error covariance of the model errors, and R is the error covariance of the observations. Clearly, if $\alpha = 1$ we find the minimum back. We choose the scalar α_i such that the weights are equal, leading to

$$\alpha = 1 - \sqrt{1 - b_i/a_i} \tag{52}$$

in which $a_i = 0.5x_i^T R^{-1} HKx$ and $b_i = 0.5x_i^T R^{-1} x_i - C - \log w_i^{rest}$. Here $x = y^n - Hf(x_i^{n-1})$, C is the chosen target weight level, and w_i^{rest} denotes the relative weights of each particle i up to this time step, related to the proposal density explained above.

Of course, this last step towards the observations cannot be fully deterministic. A deterministic proposal would mean that the proposal transition density q can be zero while the target transition density p is non zero, leading to division by zero, because for a deterministic move the transition density is a delta function. The proposal transition density could be chosen a Gaussian, but since the weights have q in the denominator a draw from the tail of a Gaussian would lead to a very high weight for a particle that is perturbed by a relatively large amount. To avoid this q is chosen in the last step before the observations as a mixture density

$$q(x_i^n|x_i') = (1-\gamma)U(-a,a) + \gamma N(0,a^2) \tag{53}$$

in which x_i' the particle before the last random step, and γ and a are small. By choosing γ small the change of having to choose from $N(0,a^2)$ can be made as small as desired. For instance, it can be made dependent on the number of particles N.

To conclude, the almost-equal-weight scheme consists of the following steps:

1 Use the modified model equations for each particle for all time steps between observations.

2 Calculate, for each particle i for each of these time steps

$$w_i^n = w_i^{n-1} \frac{p(x_i^n|x_i^{n-1})}{q(x_i^n|x_i^{n-1}, y^m)} \tag{54}$$

3 At the last time step before the observations calculate the maximum weights for each particles and determine $C = -\log w_{target}$

4 Determine the deterministic moves by solving for α_i for each particle as outlined above.

5 Choose a random move for each particle from the proposal density (53).

6 Add these random move to each deterministic move, and calculate the full posterior weight.

7 Resample, and include the particles that have been neglected from step 4 on.

Finally, it is stressed again that we do solve the fully nonlinear data assimilation problem with this efficient particle filter, and the only approximation is in the ensemble size. All other steps are completely compensated for in Bayes Theorem via the proposal density freedom.

4.6 Application to the barotropic vorticity equations

Here a few results using the new particle filter with almost equal weights are shown. Figure 4 shows the application of the method to the highly chaotic barotropic vorticity equation, governed by:

$$\begin{aligned} \frac{\partial q}{\partial t} - \frac{\partial \psi}{\partial y}\frac{\partial q}{\partial x} + \frac{\partial \psi}{\partial x}\frac{\partial q}{\partial y} &= \beta \\ q = \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} & \end{aligned} \tag{55}$$

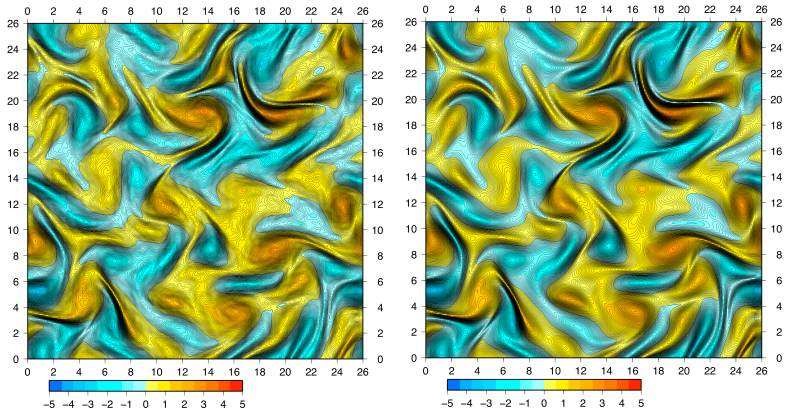

Figure 4: Snap shot of the vorticity field of the truth (right) and the particle filter mean (left) at time 25. Note the highly chaotic state of the fields, and the close to perfect tracking.

in which q is the vorticity field, ψ is the streamfunction, and β is a random noise term representing errors in the model equations. It was chosen from a multivariate Gaussian with mean zero, variance 0.01, and decorrelation lengthscale 4 gridpoints. The equations are implemented on a 256 X 256 grid, using a semi-Lagrangian scheme with time step $\Delta t = 0.04$, grid spacing $\Delta x = \Delta y = 1/256$, leading to a state dimension of close to 65,000. The vorticity field was observed every 50 time steps on every gridpoint. The decorrelation time scale of this system is about 25 time steps, so, even though the full state is observed, this is a very hard highly nonlinear data-assimilation problem. The observations were obtained from a truth run and independent random measurement noise with standard deviation 0.05 was added to each observation.

Only 24(!) particles were used to track the posterior pdf. In the application of the new particle filter we chose $K = 0.1$ in the nudging term (except for the last time step before the new observations, where the 'almost equal weight' scheme was used, as explained above), multiplied by a linear function that is zero half way the two updates and growing to one at the new observation time. The random forcing was the same as in the original model. This allows the ensemble to spread out due to the random forcing, and pulling harder and harder towards the new observation the closer to the new update time.

Figure 4 shows the difference between the mean and the truth after 50 time steps, and figure 5 the ensemble standard deviation compared to the absolute value of the mean-truth misfit. Clearly, the truth is well represented by the mean of the ensemble. Figure 5 shows that although the spread around the truth is underestimated at several locations, it is over estimated elsewhere,

Finally, figure 6 shows that the weights are distributed as they should: they display small variance around the equal weight value 1/24 for the 24 particles. Note that the particles with zero weight had too small weight to be included in the almost equal weight scheme, and will be resampled from the rest.

Because the weights vary so little the weights can be used back in time, generating a smoother solution for this high-dimensional problem with only 24 particles.

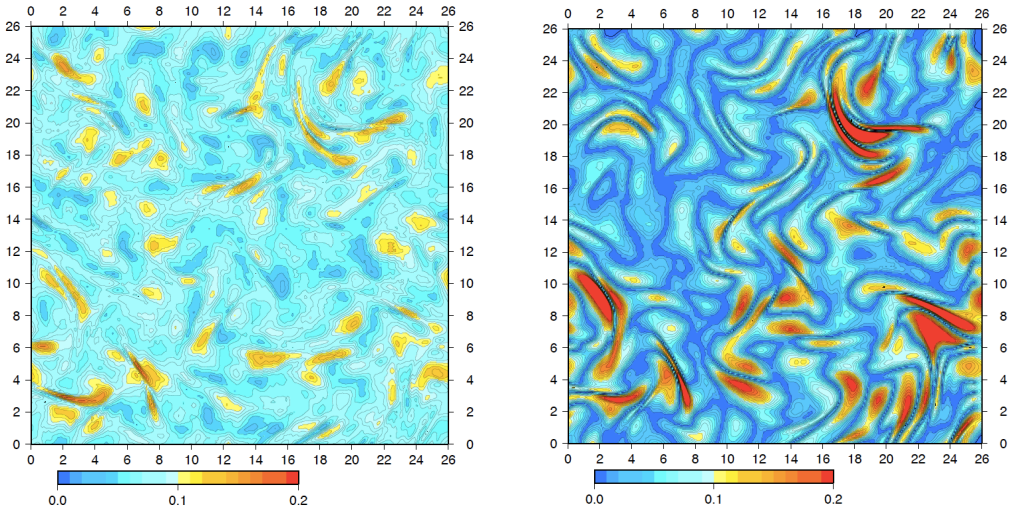

Figure 5: Snap shot of the absolute value of the mean-truth misfit and the standard deviation in the ensemble. The ensemble underestimates the spread at several locations, but averaged over the field it is slightly higher, 0.074 versus 0.056.

5 Summary and conclusions

A new particle filter has been introduced that exploits the proposal density and allows small ensemble sizes on very large dimensional problems. It was demonstrated here on the highly nonlinear 65,000 dimensional barotropic vorticity equation that simulates ocean eddy processes.

The big advantage of this method is the enormous freedom in the two steps that make up the new method. The first adds terms to the model equations that force the model towards the future observations. The simple additive terms allow easy implementation in any simulation code for atmosphere of ocean, or more general any computer code that simulates a Markov process. But also more sophisticated proposals can be used, like e.g. a weak-constraint 4DVar solution on each particle, or an Ensemble Kalman filter. The second crucial step allows the weights to be almost equal. Without this step the particle filter would still be degenerate with a large number of independent observations in the present settings. Also here a large freedom exists in how this term is implemented. We replaced the search for the intersection of a hyperplane and the pdf in the 65,000 dimensional space by a simple line search, but many other possibilities can be explored.

Acknowledgements

This work was funded by the National Environmental Research Council (NERC) directly on grant NE/H008853/1 and via the National Centre for Earth Observation (NCEO).

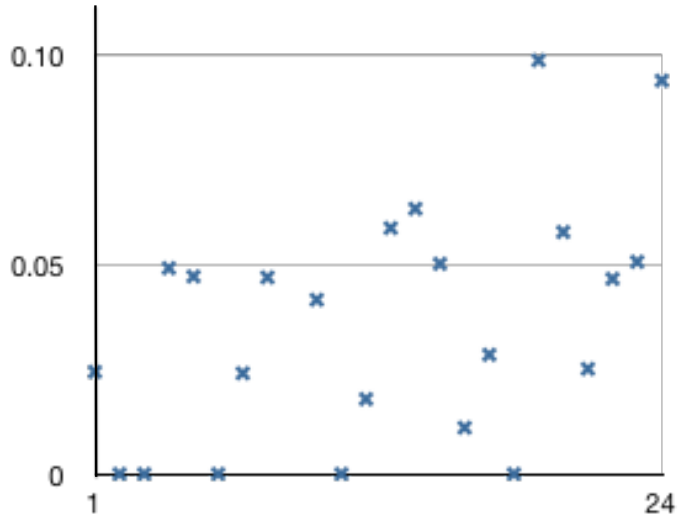

Figure 6: Weights distribution of the particles before resampling. All weights cluster around 0.05, which is close to 1/24 for uniform weights (using 24 particles). The 5 particles with weights zero will be resampled. Note that the other particles form the smoother estimate.

References

Bocquet M, Pires CA,Wu L. 2010. Beyond gaussian statistical modeling in geophysical data assimilation. Monthly Weather Review, 138, 2997-3023.

Chorin AJ, Tu X. 2009. Implicit sampling for particle filters. Proceedings of the National Academy of Sciences 106(41), 17 249-17 254.

Doucet A, de Freitas N, Gordon N. 2001. Sequential monte-carlo methods in practice. Springer-Verlag.

Gordon NJ, Salmond DJ, Smith AF. 1993. Novel approach to nonlinear/non-gaussian bayesian state estimation. IEE proceedings 140, 107-113.

Snyder C, Bengtsson T, Bickel P, Anderson J. 2008. Obstacles to high-dimensional particle filtering. Monthly Weather Review 136, 4629-4640.

Van Leeuwen PJ. 2009. Particle filtering in geophysical systems. Monthly Weather Review 137, 4089-4114.

Van Leeuwen PJ. 2010. Nonlinear data assimilation in geosciences: an extremely efficient particle filter. Quarterly Journal of the Royal Meteorological Society 136, 1991-1999.

Van Leeuwen PJ. 2011. Efficient nonlinear data-assimilation in geophysical fluid dynamics. Computers and Fluids 46, 52-58.